Stream and Feature Acquisition Visualization

Aditya Ganesh Khedekar aditya.khedekar@st.ovgu.de
Otto von Guericke Universität
Magdeburg, Sachsen Anhalt, Germany

Mallika Manam mallika.manam@st.ovgu.de
Otto von Guericke Universität Magdeburg
Magdeburg, Sachsen Anhalt, Germany

Shreeya Channappa Yogesh shreeya.channappa@st.ovgu.de
Otto von Guericke Universität Magdeburg
Magdeburg, Sachsen Anhalt, Germany

Abstract

Stream data visualization is the visualization of continuously generated and real-time processed data. Stream data is of dynamic nature and the underlying concepts and the data distributions change over time, this phenomenon is called a drift. Model accuracy diminishes over time due to drifts, resulting in increasingly unreliable predictions. Detecting and visualizing these drifts is crucial for monitoring model performance providing immediate insights and enabling prompt action which is invaluable for various industries. Currently, stream data processing and visualization are active topics of research (1), focused on developing advanced techniques to handle the constantly evolving nature of data streams and to enhance real-time decision-making capabilities. In this paper, we aim to detect and visualize different types of drifts. We also present different visualization techniques to visualize missing data in streams and the type of missingness and explore multiple methods to visualize the velocity in a stream. Additionally, we also investigate visualization methods to compare the performance metrics of different learning strategies over the course of the stream. We have implemented various visualization techniques to demonstrate all the aspects of data streams as mentioned above. Our implementations are integrated into a Python package using an object-oriented approach, ensuring usability and extensibility for future study and practical applications.

Keywords: concept drift, stream velocity, data drift, learning strategies.

1 Introduction

Any sequence of data that is generated over time is referred to as a data stream. The data stream comprises a vast amount of data at diverse speeds (6). A typical challenge when analyzing data streams is that they are not always static, as the underlying data distribution changes over time (7).

Stream visualization is a visualization technique for graphically depicting data that flows over time. This helps to analyze and monitor the stream data by visualizing the patterns or trends that may occur in the data streams.

This paper addresses the topics introduced below.

1.1 Data Drift and Concept Drift

The change in data distribution over time is referred to as Data Drift. However, the change in concept over time is referred to as Concept Drift.

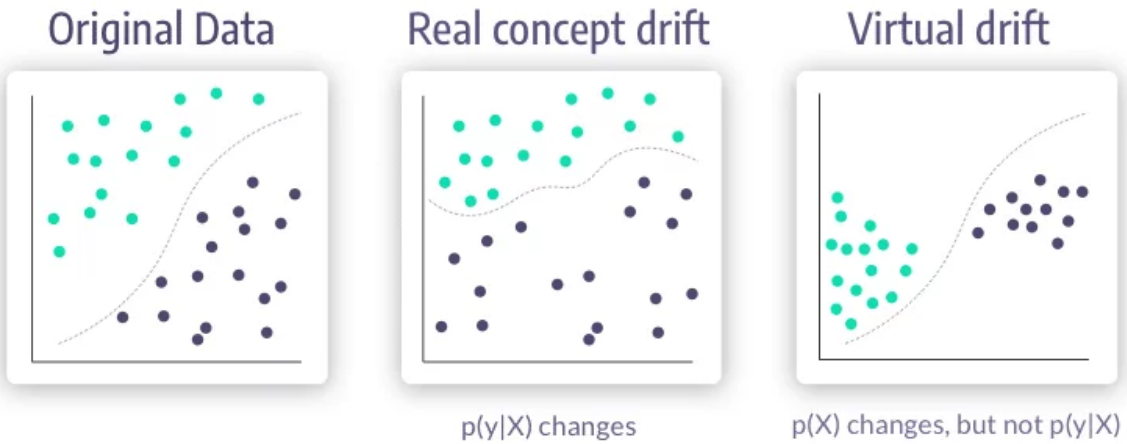

Figure 1: Types of drifts: circles represent instances; different colors represent different classes (8)

Let's say that we are working on a fraud detection system for an online payment gateway. Initially, most transactions were processed via PCs, but recently mobile devices have also become increasingly popular. Due to the finite number of mobile transactions, it is challenging for the model to distinguish between fraudulent and legitimate or legal mobile transactions, which are characterized by their frequency and lower value.

Data drift can be observed here, i.e. a change in the distribution of input parameters such as "device type" or "transaction amount", as the number of mobile transactions increases. This can affect the performance of the model. However, the core idea that defines fraud has not changed.

On the other hand, a concept drift can occur when a new fraud tactic exclusively targets mobile users, resulting in a sudden spike in fraudulent mobile transactions (see Fig. 1).

The shift or change in concept or data distribution could be categorized into Linear, Gradual, and Sudden (6) (Fig. 2). A Sudden or Abrupt drift occurs when there exists a sudden change in distribution from one time point to another time point. Where as Gradual drift occurs when there is a gradual or slow

1.2 Missing Data

Missing data refers to the absence of data values in a data set. There are various reasons why data may be missing, and this can lead to inefficient data analysis. 2

There can be trends or patterns in the way the data is missing or unavailable. Accordingly, missing data can be categorized into 3 types.

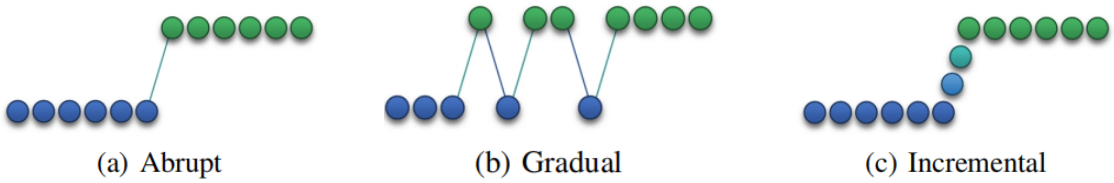

Figure 2: Figures depicting Abrupt, Gradual, and Incremental Drifts (3)

1. MAR (Missing At Random) - Depends on data that are observed, but not on missing(unobserved) data itself (9).
    - Eg., Two variables age and income.
    - Income is missing for very 'young' people, and not due to income itself.
2. MCAR (Missing Completely At Random) - Relies neither on observed nor on the data that is not seen (unobserved) (9).
    - Eg., Missing values of overdue books in library due to human error (forgot to type).
    - Missing values are not related to any other variable/data in system.
3. 3. MNAR (Missing Not At Random) - The missing data are related to the unobserved data.
    - Eg., Income or weight.
    - People with high income or high weight are less likely to report it to a survey.

1.3 Data Velocity over Time

Data velocity over time refers to the rate at which data instances are arrived at or generated at a given time. It quantifies how frequently data arrives and can help to understand the underlying dynamics of the data flow in a system. For example, in a streaming data setting, data velocity can indicate the number of records processed per second.

1.4 Learning Strategies

Learning strategies are the different models or methods that are applied to batches of a dataset to achieve the best possible performance. It is also important to visualize the results of multiple strategy evaluations to determine the most effective model for each batch. Visualizations enable quick comparisons of model performance across batches, making it simpler to see which models are yielding good results.

2 Related Study

2.0.1 Visualization of Streaming Data: Observing Change and Context in Information Visualization Techniques

This paper discusses the importance of having incremental visualization techniques for data streams. Several existing methods are reviewed and analyzed in terms of the potential loss of temporal and relational context between new and old data, along with the advantages, disadvantages, and future work.

The first method discussed is treemap – a popular technique used to depict hierarchical data where each rectangle represents the attribute(s) of a node. Treemap when applied to streaming data, is used to show the new data in the context of the past data where the color and size of the rectangle can be used to show the recent changes in the data (2).

The next visualization technique mentioned is the scatter plot – used to represent multi-dimensional data in the cartesian space and also used to find correlations between attributes. Here each data point is visualized independently of other objects. The opacity and color can be used to encode the time(age) of the data object (2). Minimal loss of context happens when the arriving data point is out of range.

Stream graphs are discussed next, also called stacked graphs or themeriver. It shows several streams(variables) that change over time and are layered on top of each other along the timeline. Here the layers that change the least are placed in the middle whereas the attributes that change the most are placed in the upper and lower layers (2). This visualization may not be the best for data streams as the whole visualization has to be recomputed whenever new data arrives.

Horizon graphs are explored next, it is a visualization technique mostly used to display multiple time series due to its efficient use of space and color (2). It can be used for stream data as well, but there is a loss of context when the new data is out of range and when the order of the horizon charts changes.

The next visualization method is line chart, which when applied to stream data is similar to stream graphs and has the same re-scaling issue when the data is out of range. It also suffers from disadvantages like over-plotting and clutter.

The next technique explored is pixel-oriented visualizations which are mainly used for multi-dimensional data. The main idea here is to use a pixel to represent an attribute and use the color to represent the value. The arrangement of the pixels is dependent on different criteria and can be user-defined. The disadvantage is that the pixels have to be recolored if the data value is out of range for the chosen colormap.

The last visualization technique is word clouds to visualize text data. The words are packed tightly together in two ways - force-directed layout and spiral. There is significant context loss in the former, as new words require the entire layout to be recomputed. Whereas in the latter, the highest-weight word is placed at the center and the rest are arranged in a spiral, maintaining context better, only losing it if updates are substantial.

2.0.2 McDiarmid Drift Detection Methods for Evolving Data Streams

This paper presents the McDiarmid Drift Detection Method (MDDM) which is a novel method to detect concept drifts in a classification setting. This method leverages the McDiarmid inequality to detect drifts in evolving data streams. A sliding window is slid over

the prediction outcomes and each outcome is assigned a weight such that recent entries are given more importance over other entries in the window (3). The paper introduces three weighting schemes - arithmetic, geometric, and Euler. The weighted mean of the current window is compared with the highest window mean observed so far in the MDDM algorithm. A significant difference between these weighted means signals a concept drift, as determined by the McDiarmid inequality.

Through comprehensive testing on both synthetic and real-world data streams, the novel method introduced in this paper has demonstrated superior performance compared to the other state-of-the-art methods. MDDM achieves shorter detection delays, reduces false negative rates, and maintains high classification accuracy.

2.0.3 Change Point Detection and Dealing with Gradual and Multi-Order Dynamics in Process Mining

This paper addresses the issue of detecting concept drifts in process mining, where processes change with time, and introduces novel methods to detect sudden and gradual drifts, as well as multi-order dynamics, which are changes occurring at multiple time scales.

The methodology used here is to use a statistical test to compare data populations to detect a drift with the help of an adaptive window, where the window size is changed dynamically to avoid false positives and negatives. In the adaptive window approach, there is a maximum and a minimum window size along with a step size. The window starts with the minimum size and if no change is detected then the window size is increased by the step size until it reaches the maximum, if there is still no change then the old data is replaced by new ones. If a change is detected then a new window is created starting after the change point detected (3).

The change detection algorithm is used to detect sudden drifts. According to the algorithm, the window is divided into two halves, and a statistical test is used to determine a change. If the p-value is below a threshold, then it indicates that a change has occurred. If no change is detected, the window is either resized or replaced according to the adaptive window algorithm. When a change is detected, the algorithm recursively bisects the window until the exact location of the change point is located.

A different windowing technique is used to detect gradual drifts. Since gradual drifts take place slowly with one concept gradually fading out while the other one gradually takes over, the authors propose separating the data populations with intentional gaps (3). Adaptive windowing and change point detection algorithms are applied to find the exact points of change. Since gradual change occurs over a time period and not suddenly, the technique of introducing a gap in the window can be beneficial to achieve low p-values, therefore enhancing the ability to detect gradual drifts accurately even when the transition between process states is subtle.

2.0.4 Detecting Drifts in Data Streams using Kullback-Leibler (KL) divergence measure for data engineering applications

In this paper, the statistical measure - Population Stability Index (PSI), as an extension of Kullback-Leibler (KL) divergence is used to detect and quantify data drifts. It discusses how PSI is an effective method to detect and mitigate drifts as compared to KL divergence.

KL Divergence is a statistical measure that quantifies the dissimilarities between data distributions. However, it is not a true distance measure as its definition is not symmetric (4). The calculation of PSI is derived from the KL Divergence formula but modified to provide a symmetrical measure. The paper also has provided guidelines interpreting data drift with PSI values: PSI < 0.1 indicates no significant drift, $0.1 \leq$ PSI < 0.2 indicates moderate drift, and PSI ≥ 0.2 indicates significant drift (4).

3 Implementation and Visualizations

We have applied the specified objectives using three distinct datasets: 'cfpdss.csv', 'cfpdss_m0.5.csv', and 'experiment.csv'.

- 'cfpdss.csv': This is a synthetically generated dataset designed to simulate stream data. It includes 10 features, comprised of 5 numerical and 5 binary categorical variables. The target variable is also binary categorical.
- 'cfpdss_m0.5.csv': This dataset is identical to 'cfpdss.csv' but contains missing values.
- 'experiment.csv': This dataset encompasses seven different strategies or models, and is divided into batches, each containing approximately 50 instances. Each strategy/model is evaluated using a kappa score for every batch, enabling a comparative analysis of performance over time.

These datasets enable a comprehensive examination of our objectives, facilitating the assessment of concept drift detection in various data scenarios, including handling missing data and evaluating multiple models.

3.1 Data Drift Detection and Visualization

Data drift occurs when there is a change in the data distribution. To identify data drift, we apply a sliding window mechanism over the data and conduct statistical tests to detect changes in the data distribution within each window. For numerical features, the Kolmogorov-Smirnov (KS) test is employed to identify drifts. For categorical features, we use the Population Stability Index (PSI) to measure changes in distribution.

The KS test is a non-parametric test that is used to compare the cumulative distribution functions (CDFs) between two samples. CDFs describe the probability that a random variable falls within a certain range. KS Test compares the two CDFs and identifies if the two samples come from the same distribution or if there are significant differences between them. A KS statistic is calculated, denoted by D, which quantifies the maximum difference between the CDFs of the two samples. The formula for calculating the KS statistic – D, is as follows:

$$D = max_x|F_1(x) - F_2(x)|$$

Here, F1(x) and F2(x) denote the CDFs of the first and the second sample at any given point x. The absolute value of the difference between the two functions is taken across the entire range of the data and the maximum of these differences constitutes the KS statistic. The null hypothesis would be that both the samples come from the same distribution or

that there is no difference between the two samples. The p-value is calculated with the KS statistic and the sample size of both the distribution in consideration. If the p-value is less than the significance level of 1% or 5%, then we reject the null hypothesis.

PSI is a symmetric metric that measures the relative entropy, or difference in information represented by two distributions. It is similar to measuring the distance between the two data distributions quantifying the difference in the distribution. Given two probability distributions P and Q of a discrete random variable x, PSI is given as follows:

$$PSI = \sum_i (P(x_i) - Q(x_i)) \ln \frac{P(x_i)}{Q(x_i)}$$

PSI thresholds are used to determine the similarity between samples. PSI less than 0.1 is considered similar or no significant difference. PSI between 0.1 and 0.2 is regarded as a substantial divergence and PSI > 0.2 is considered a significant difference between the distributions (4). KS test is not preferred for categorical data because KS test compares the CDFs of the two samples which is defined only for continuous or numerical data. Therefore, PSI is used for categorical data as PSI is designed to compare the proportions of categories in the two samples which is suitable for categorical data.

As mentioned earlier, we use a sliding window approach, where a window of size 'n' is moved across the dataset. This window is divided into two halves, and statistical tests (PSI or KS test) are performed to check for changes in the data distribution. For numerical features, the KS test is applied, and if the resulting p-value is less than the significance level (here 0.001), it indicates a drift. For categorical features, the PSI test is used, and if the PSI value exceeds a predefined threshold (here 0.12), a drift is detected. Once these conditions are met, further analysis is conducted to determine the specific type of drift that has occurred.

- A sudden drift occurs when the absolute value of the difference between the mean of the two halves of the window is greater than the standard deviation of the complete window.

  $$suddendrift \rightarrow abs(mean_{diff}) > std(window)$$

  $$mean_{diff} = mean(second\ half\ of\ the\ window) - mean(first\ half\ of\ the\ window)$$

- Linear drift is detected when the difference between the mean of the two halves of the window is greater than 0.

  $$mean_{diff} > 0$$

- For the detection of gradual drift, we change the windowing technique, we introduce a gap of a specific size in-between the two halves of the windows. Gradual drift occurs when the old concept gradually fades out and the new concept starts. To capture this, we have the gap so that lower p-values and higher PSI values can be achieved.

A scatter plot with a moving average is used to visualize data drifts for both numerical and categorical features as shown in Fig 3 and Fig 4. The x-axis represents time points, while the y-axis represents the range of values taken by that particular feature. Vertical bands

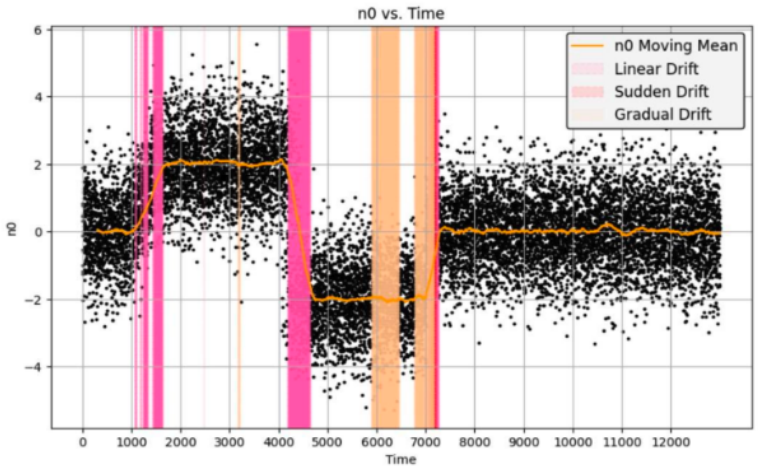

Figure 3: Graph depicting sudden, linear, and gradual drift with window size = 300 and gap size = 100 for the numerical feature 'n0'.

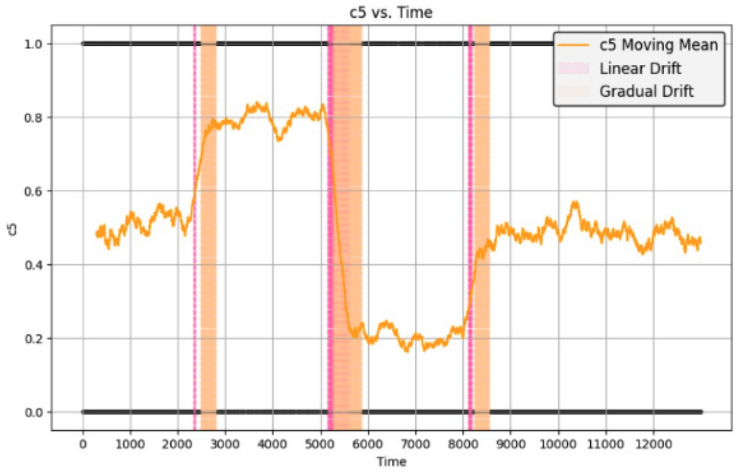

Figure 4: Graph depicting linear and gradual drift for the categorical feature 'c5' with window size 300 and gap size 100.

of different colors indicate various types of drifts, illustrating the specific time points at which they occur. This method provides a clear depiction of the temporal changes in data distribution, facilitating the identification and analysis of data drifts over time.

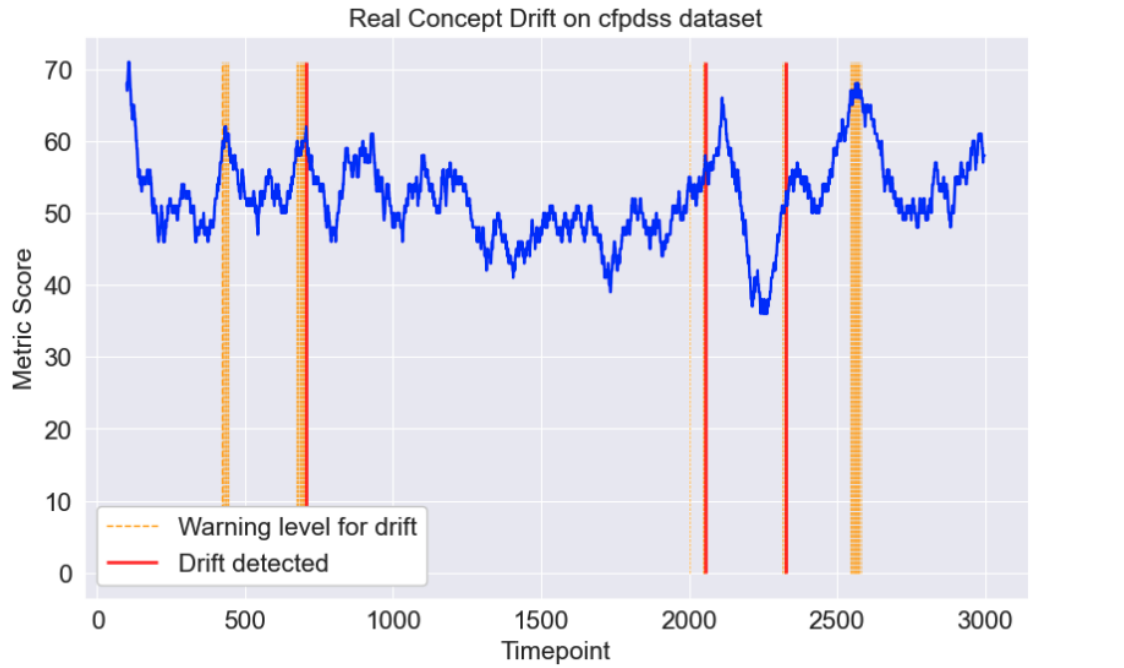

Figure 5: Image depicting concept drift along with warning levels

3.2 Concept Drift Detection and Visualization

McDiarmid Drift Detector (MDDM) is used to detect concept drifts over the data stream. This method utilizes the McDiarmid inequality to detect drifts. This approach makes use of a sliding window and a weighting scheme on the prediction results such that higher weights are assigned to recent entries in the sliding window. The detection algorithm monitors instances by comparing the weighted mean of elements within a sliding window to the highest weighted mean recorded to date. If the difference between these two weighted means is substantial as per the McDiarmid inequality, it indicates a concept drift. There are three different weighting schemes – arithmetic, geometric, and Euler scheme.

According to the MDDM algorithm, a window of size n is slid over the prediction results and 1 is inserted if the prediction is correct else 0. A weight is associated with every element in the window such that $w_i < w_{i+1}$ (3). The weighted average μ_{wt} and the maximum weighted average μ_{wm} so far are calculated as the inputs are processed. A significant difference between μ_{wt} and μ_{wm} indicates a drift (3).

$$\Delta\mu = \mu_w^t - \mu_w^m \geq \epsilon_d \rightarrow Drift \ := \ True$$

The McDiarmid's inequality is leveraged to check if the difference is substantial. For a given confidence level δ_M , the value of ϵ_M is given by:

$$\epsilon_M = \sqrt{\frac{\sum_{i=1}^{n} c_i^2}{2} \ln \frac{1}{\delta_M}}$$

Given the confidence level δ_w, the McDiarmid inequality detects a drift if $\Delta\mu_w \geq \epsilon_w$, where

$$\epsilon_w = \sqrt{\frac{\sum_{i=1}^{n} v_i^2}{2} \ln \frac{1}{\delta_w}}$$

Where vi is given by

$$v_i = \frac{w_i}{\sum_{i=1}^{n} w_i}$$

In our implementation, the Hoeffding Tree classifier is trained incrementally on the dataset, and concept drift detection is handled using MDDM with an arithmetic weighting scheme. We utilize a sliding window approach implemented through a deque to maintain a window of the most recent predictions and their actual outcomes. This window is used to calculate the performance metric, which in this case is accuracy. For each prediction, if the predicted output matches the actual output, a value of 1 is added to the MDDM model, and a 0 is added otherwise. We set the significance level at 0.005 to detect warnings and at 0.001 to confirm concept drift. Warning levels act as an early alert mechanism, indicating potential changes in the data distribution before a full concept drift is confirmed. These levels signal that the model should begin preparing for possible adjustments in response to the detected changes.

For visualization, a line graph illustrating the performance of a Hoeffding Tree classifier, monitored for concept drifts over time is used. The X-axis represents time points, and the Y-axis depicts the accuracy metric, expressed as a percentage. Various vertical lines indicate key events: orange lines mark warning levels signaling potential changes in the data distribution, while red lines highlight confirmed concept drifts as shown in Fig 5.

3.3 Missing Data Visualization and Type of Missingness ()

3.3.1 Stacked Bar Graph

We have used a stacked bar chart to show the proportions of available and missing data. A stacked bar chart is essentially a bar chart with multiple bars stacked on top of each other.

The figure 6 represents visualization of a categorical variable. The x-axis is considered as a time period, with each period consisting of 1000 instances or records. And the y-axis is represented as the frequency of the labels “A”, “B” and “Missing”. For instance, half of the bar is colored green, which means that about 50% of the data in c5 is missing for period 0, the first 1000 instances.

3.3.2 Interactive HeatMap plot for Missing Data Visualization

HeatMap is a visualization technique which we have used to represent the values in different colors. In the figure 7, x-Axis represents features of the dataset and y-axis indicates the time points at which instances arrived.

Key features include:

Interactive Slider - Plot based on the user selection of start and end time point sliders.

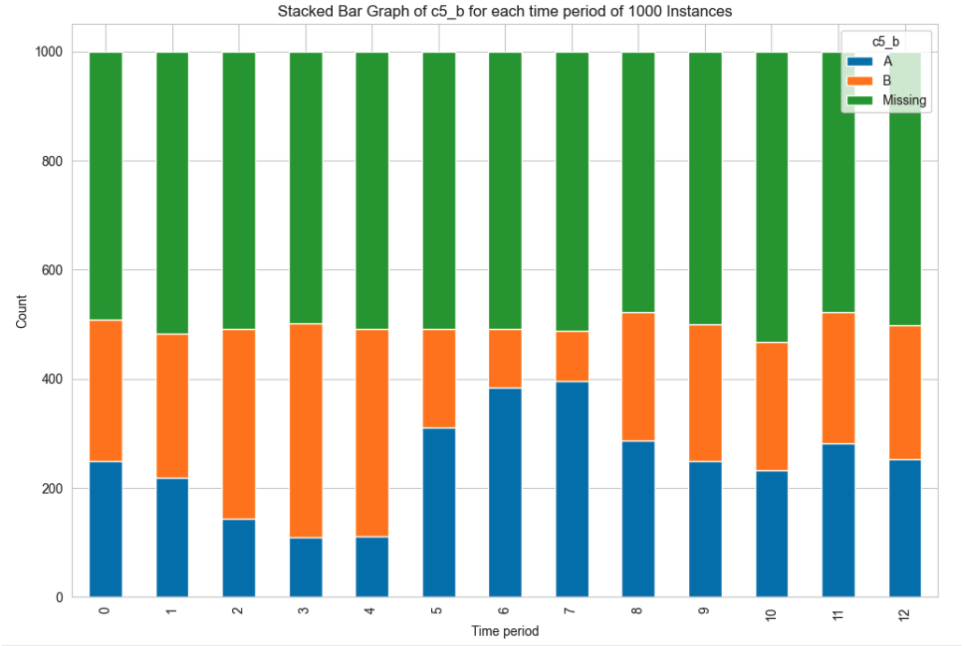

Figure 6: Stacked bar chart depicting the number of missing data in the categorical feature 'c5'

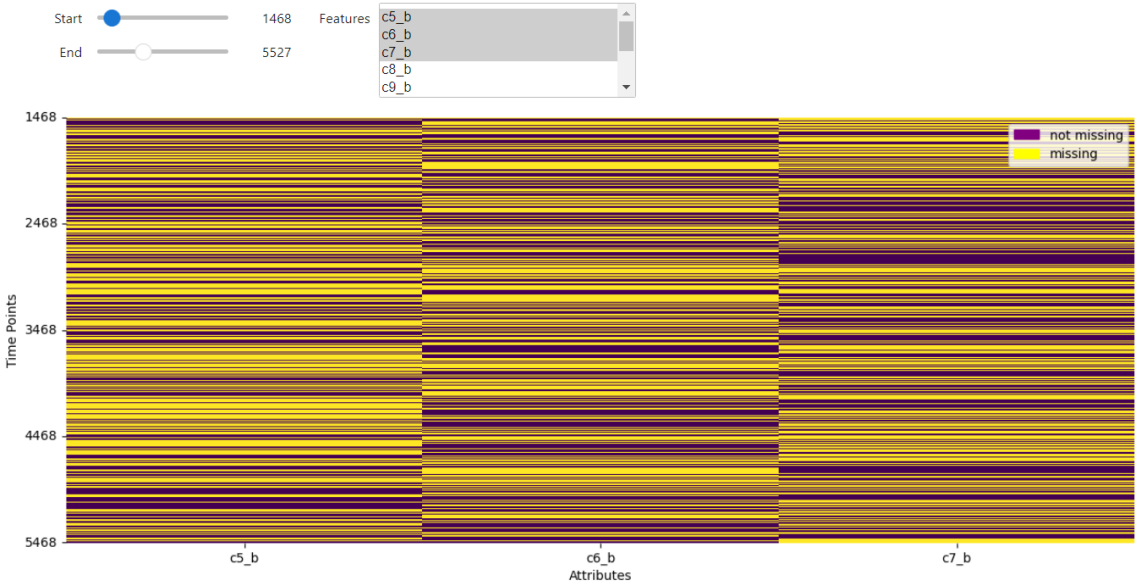

Figure 7: Interactive HeatMap

Multiple Feature Selection and Visualization - Ability to select which 2 or 3 features the user can compare.

Easily switch between the standard plot method and the interactive display method based on user preference.

3.3.3 Scatter Plot

To visualize missing and non-missing values in a dataset, a scatter plot was employed using both the complete dataset and the dataset with missing values. For numerical features, all data points were plotted, and different colors were utilized to distinguish between present and missing data points. Specifically, if a data point was missing in the incomplete dataset, it was plotted in a distinct color as shown in Fig 8.

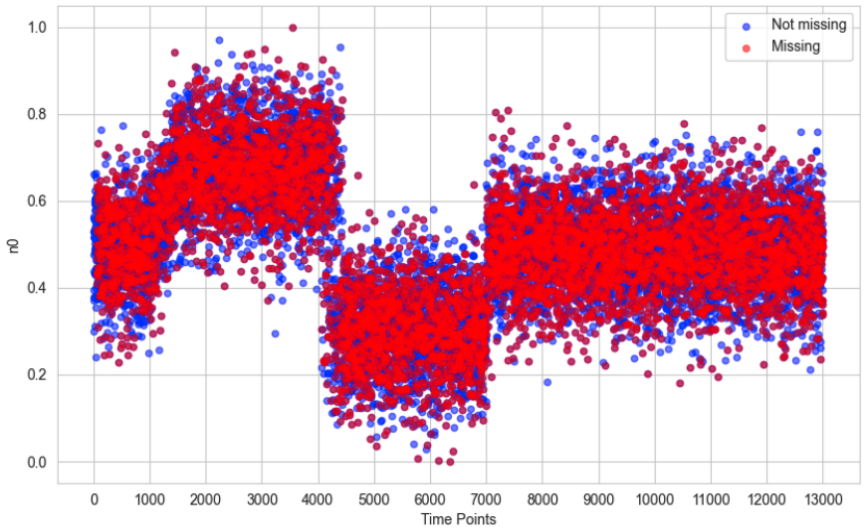

Figure 8: Scatter plot depicting missing and non-missing values for the numerical feature 'n0'.

The same methodology was applied for categorical features however, missing values were adjusted by subtracting 0.5 so that there is a clear distinction between the missing and non-missing values and that they do not appear in parallel lines as the categorical features have binary values as displayed in Fig 9.

3.3.4 HeatMap Plot for Type of Missingness

Missing data can fall into three categories: Missing At Random (MAR), Missing Completely At Random (MCAR), and Missing Not At Random (MNAR). When data is missing at random (MAR), the absence of values in a feature is dependent on another feature. Statistical tests are performed to identify such dependencies or correlations among features. We have used the Chi-Square test, which helps determine if the missing values in one feature are related to another feature. The Chi-Square test determines if there is a significant association between two categorical variables. The null hypothesis here would be that there is no relationship between the two variables, If the calculated p-values are greater than the

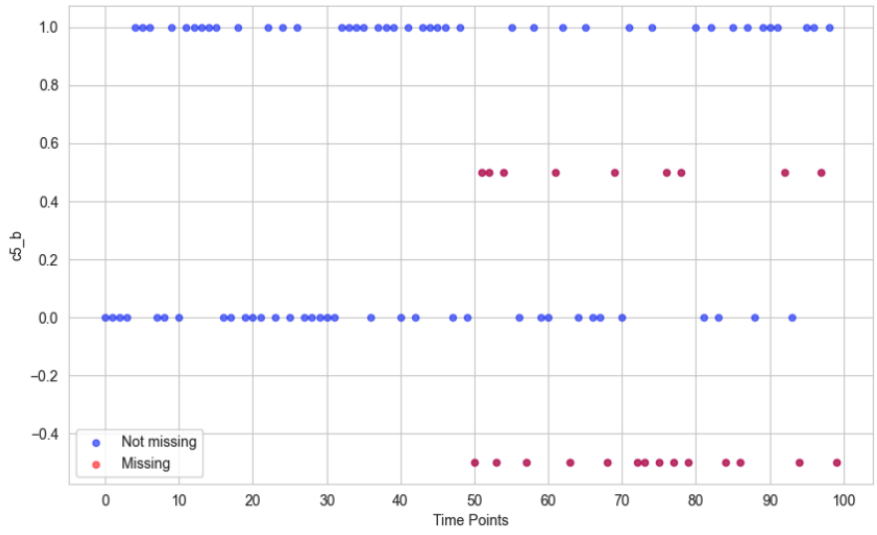

Figure 9: Scatter plot depicting missing and non-missing data in the categorical feature 'c5'.

significance level then we fail to reject the null hypothesis. The significance level used here is 0.05.

Since Chi-Square tests are best suited for categorical variables, we have done binning on the numerical features in our dataset with the help of decision trees. Binning for numerical values in the context of decision trees involves splitting the continuous numerical features into discrete intervals or bins in such a way that minimizes impurity and maximizes information gain.

Heat maps are used to visualize the association between the complete and incomplete features as shown in Fig 10. The p-values are also included in the heatmap so that associations where the p-value is less than the significance level can be identified easily.

3.4 Visualization of Velocity in Stream

3.4.1 Stream Graph

Instead of displaying granular information about each point in time, Stream graph helps to visualize, analyze, and identify trends or patterns over time. In the Stream graph, time is usually shown on a horizontal axis, with the latest time points on the right and older times on the left. Depending on the situation, days, months, or years can be used as time units. In our dataset, we have the times denoted as integers starting from 1 to 13000. The size of the measured value is shown on the vertical axis. This can be percentages, numbers, or any other type of quantifiable measurement.

The rate at which data is generated and processed in a system is shown in figure 11. In particular, it shows the speed of the data over time. The Stream Graph in figure 11 contains two different levels: an orange level and a blue level. These two levels illustrate the flow

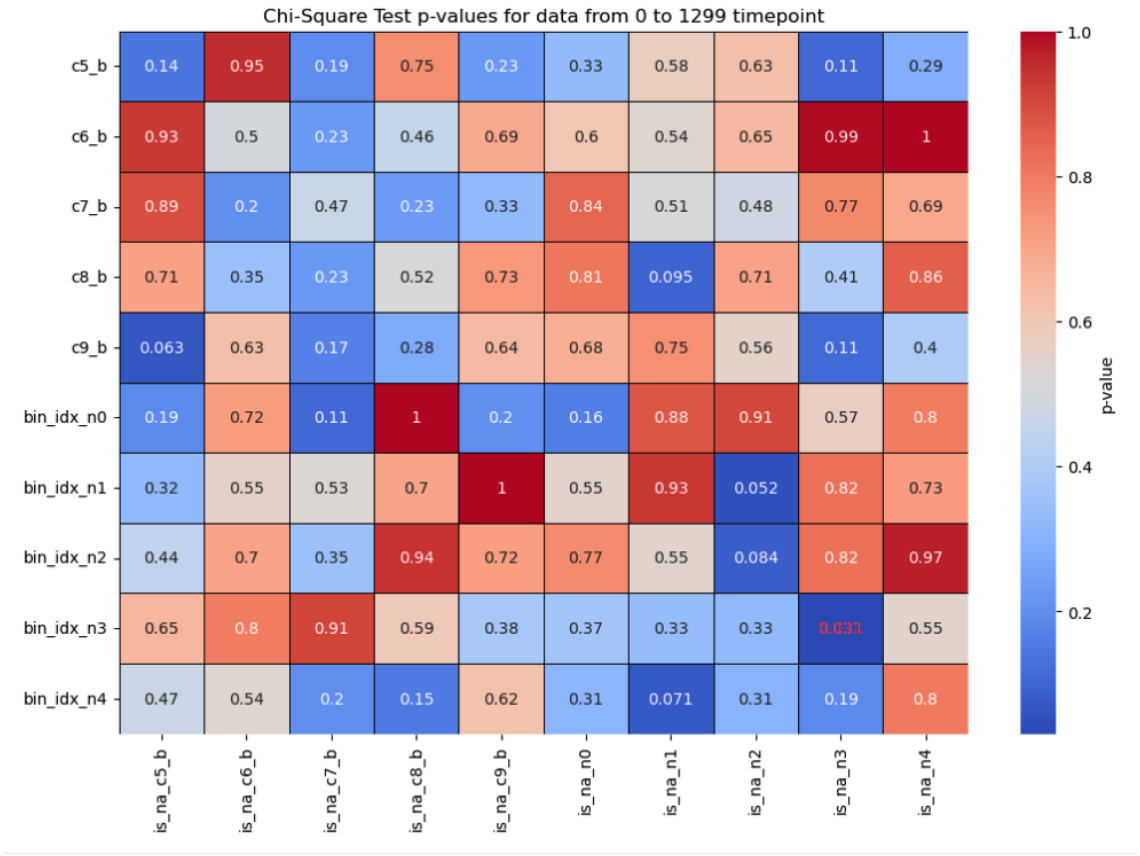

Figure 10: Heapmap with p-values from chi-square test, showing the relationship between the complete and incomplete features to check for MAR missingness.

and development of data over time. Time is represented by the x-axis, which allows us to recognize patterns and trends. The interval size of 50 used to aggregate the data points is indicated in this case by the expression "bin size = 50". It indicates how the dataset rows are divided into bins of 50 rows each.

3.4.2 Stacked Bar Graph

Another visualization approach that we considered to be effective for visualizing velocity is a Stacked bar graph which is described in the section 3.3.1

The distribution of the categorical feature "c5" is shown in the figure 12. The plot is shown over 20 periods with 10 time points per period. The speed of the feature values is indicated by the y-axis, while the x-axis indicates the time intervals. While c5_b (green) indicates the number of missing data values, c5_b 0 (blue) stands for the frequency of the presence of the value b. c5_b 1 (orange) indicates the frequency of the presence of other categorical values besides b. This makes it easier to recognize the temporal fluctuations over time and to identify any trends or patterns if exists.

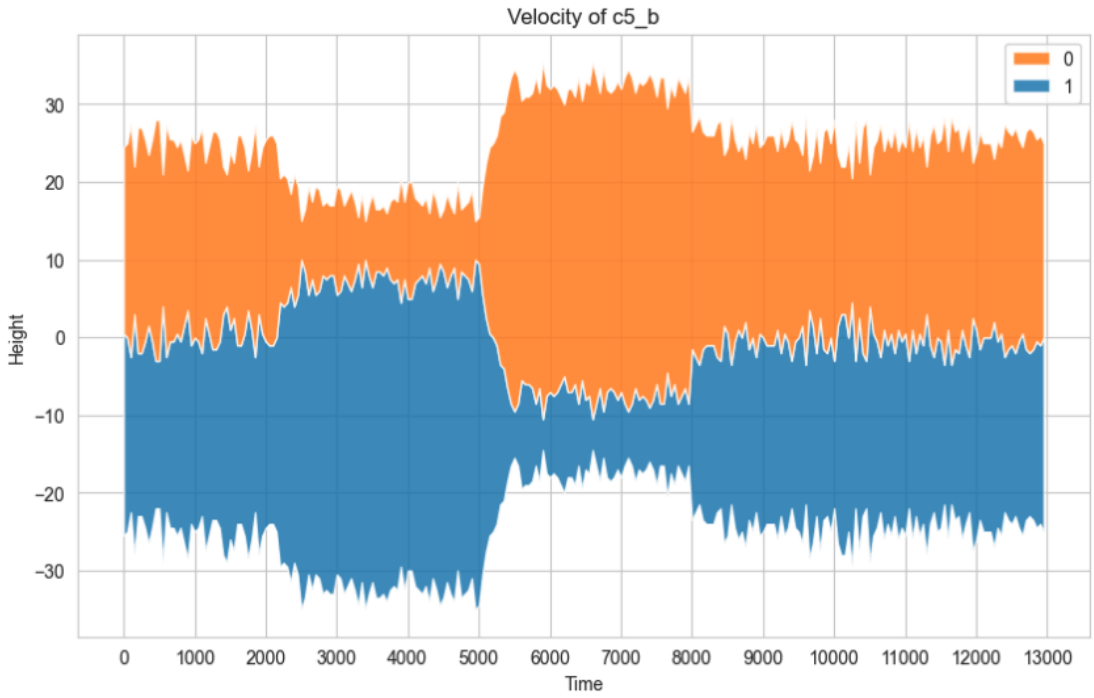

Figure 11: Stream Graph for categorical feature 'c5' with bin size = 50

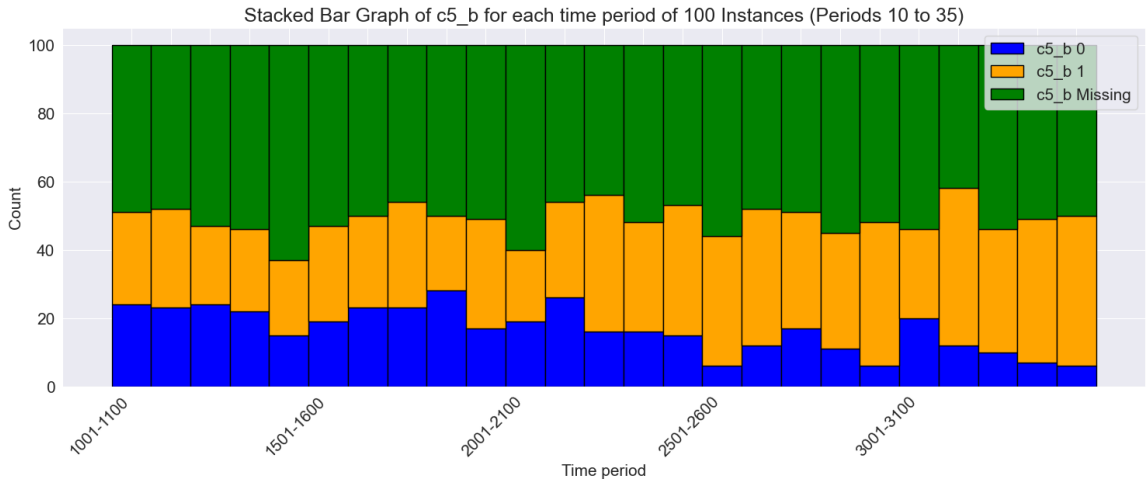

Figure 12: Stacked bar graph

3.4.3 Rolling Mean and Standard Deviation Graph

The plots in figure 13 show the rolling or moving averages and standard deviations for a given window size 10 for the numerical features n0 and n1. Multiple numeric features can be specified as input. This graph generates subplots for each feature with a common x-axis, allowing for better comparative visual analysis between features.

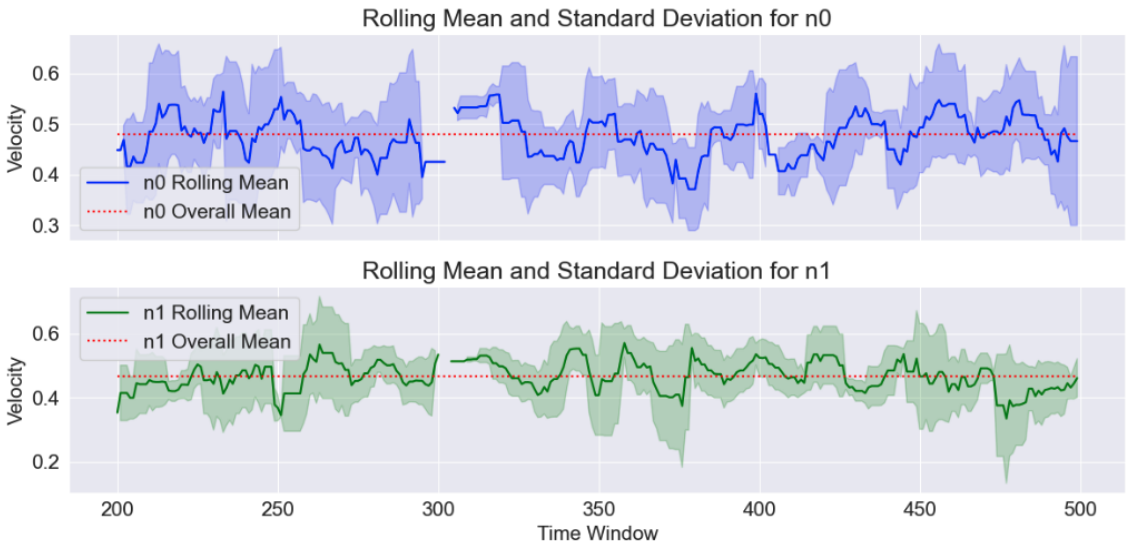

Figure 13: Moving Mean and Standard Deviations for numerical features

3.5 Learning Strategies Plot

When evaluating various models or strategies, it is common to visualize their performance for each batch. However, when the number of strategies increases, the plot becomes cluttered making it difficult to identify the best-performing strategy at a glance and it would require additional effort to discern which strategy is performing best in each batch, leading to potential oversight of key insights (refer to figure 14).

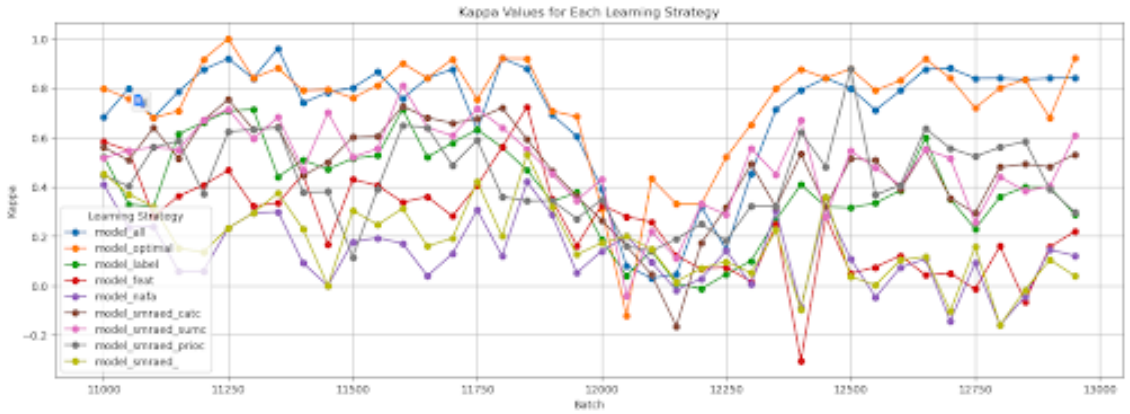

Figure 14: Visualization of Kappa values for each Learning Strategy

To address this issue, we propose a strategy plot that is similar to a line chart to indicate the winning strategy in each batch (refer figure 15). Additionally, it shows the margin or difference by which the winning strategy outperforms the second-best strategy. This enhancement not only simplifies the comparison but also highlights the most effective

strategy over time, making it easier to identify the overall winner. This refined visualization aids in quickly spotting tr ends and making informed decisions without the need for manual inspection.

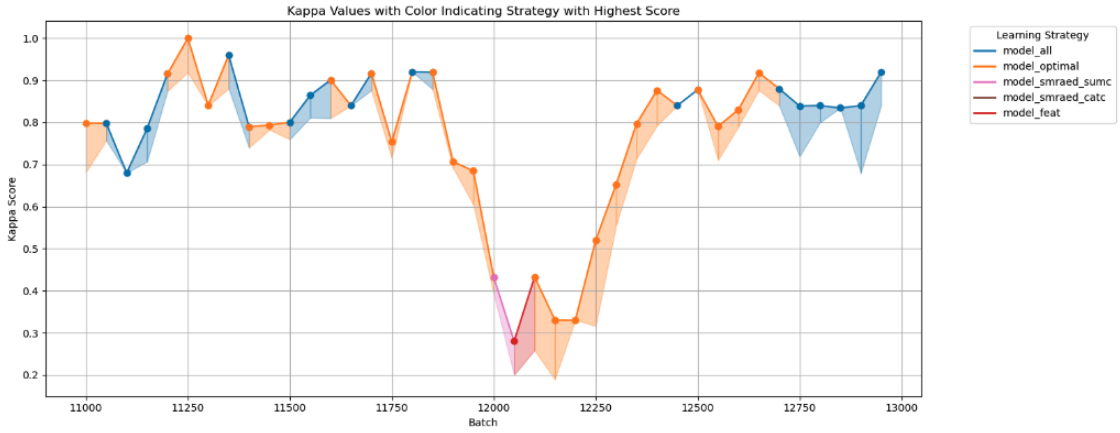

Figure 15: Learning Strategies

4 Conclusion and Future Work

It is established that in order to maintain consistent model performance and an accurate learning process, it is of utmost importance to detect and visualize drifts. It is also required to inspect different elements associated with data streams to be more efficient in evolving environments. In this paper, we have implemented drift detection and multiple visualization techniques to represent different types of drifts, data velocity, missing data, type of missingness, and lastly the performance of different learning strategies - integrated in a Python package.

As a part of future work, we aim to develop more robust drift detection techniques for sudden, linear, and gradual drifts for both data and real concept drifts. Additionally, we also plan to explore methods to identify the type of missing values and how the missingness changes over time in data streams.

4.1 Project Code

GitHub repository: `https://github.com/aditya0by0/stream-viz`.

User Guide :
`https://github.com/aditya0by0/stream-viz/blob/main/stream_viz/tutorial/UserGuide.ipynb`

Python raw package :
`https://github.com/aditya0by0/stream-viz/tree/main/stream_viz`

References

[1] M. Sarnovský, "Concept Drift Visualization Using Feature Importance on the Streaming Data," 2022 IEEE 20th Jubilee World Symposium on Applied Machine Intelligence and Informatics (SAMI), Poprad, Slovakia, 2022, pp. 000449-000454, doi: 10.1109/SAMI54271.2022.9780841.

[2] M. Krstajić and D. A. Keim, "Visualization of streaming data: Observing change and context in information visualization techniques," 2013 IEEE International Conference on Big Data, Silicon Valley, CA, USA, 2013, pp. 41-47, doi: 10.1109/BigData.2013.6691713.

[3] A. Pesaranghader, H. L. Viktor and E. Paquet, "McDiarmid Drift Detection Methods for Evolving Data Streams," 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 2018, pp. 1-9, doi: 10.1109/IJCNN.2018.8489260.

[4] Kurian, J.F., Allali, M. Detecting drifts in data streams using Kullback-Leibler (KL) divergence measure for data engineering applications. J. of Data, Inf. and Manag. (2024). https://doi.org/10.1007/s42488-024-00119-y

[5] C. Chow and C. Liu, "Approximating discrete probability distributions with dependence trees," in IEEE Transactions on Information Theory, vol. 14, no. 3, pp. 462-467, May 1968, doi: 10.1109/TIT.1968.1054142

[6] Supriya Agrahari, Anil Kumar Singh, Concept Drift Detection in Data Stream Mining : A literature review, Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 10, Part B, 2022, Pages 9523-9540, ISSN 1319-1578, https://doi.org/10.1016/j.jksuci.2021.11.006. (https://www.sciencedirect.com/science/article/pii/S1319157821003062).

[7] Heng Wang and Z. Abraham, "Concept drift detection for streaming data," 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, 2015, pp. 1-9, doi: 10.1109/IJCNN.2015.7280398.

[8] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. ACM Comput. Surv. 46, 4, Article 44 (April 2014), 37 pages. https://doi.org/10.1145/2523813

[9] SCHAFER JL,Missing data: our view of the state of the art,Psychol Methods,,,2002,7,,147-177,https://cir.nii.ac.jp/crid/1573387451195255296,